

dl_ap (/)



404 - Page not found (/404.html) About dl_ap (/aboutme/) Categories (/categories/) dl_ap (/) Tag (/tag/) Tag Index (/tags/) dl_ap (/page2/) dl_ap (/page3/)

Mạng nơ-ron tích chập - Convolutional Neural Network (CNN)

Mục lục:

- 1. Thị giác máy tính (Computer Vision)
- 2. Mạng nơ-ron tích chập (Convolutional Neural Network)
- 3. Phép chập khối
- 4. Mạng CNN một lớp
- 5. CNN đơn giản
- 6. Ví dụ một CNN cụ thể
- 7. Câu hỏi tham khảo
- 8. Tài liệu tham khảo

1. Thị giác máy tính (Computer Vision)

Nhìn vào một bức ảnh, một người với thị giác bình thường có thể dễ dàng mô tả nội dung, nhận biết và phát hiện các đối tượng được thể hiện trong bức ảnh cũng như vị trí chính xác của chúng. Tuy nhiên, việc này (đọc và hiểu một bức ảnh) khó khăn hơn nhiều đối với máy tính khi “nó” “nhìn” mỗi bức ảnh chỉ đơn thuần là một ma trận số (tập hợp các điểm ảnh - **pixel** biểu diễn dưới dạng số theo một hệ cụ thể thường là RGB (Red - Green - Blue)). Mục tiêu chính của Thị giác máy tính (Computer Vision) - một nhánh của trí tuệ nhân tạo (Artificial Intelligence) là tìm ra cầu nối giữa ma trận số này và thông tin ngữ nghĩa ẩn chứa trong ảnh. Thị giác máy tính tập trung giải quyết những bài toán như:

- **Phân loại ảnh, miêu tả ảnh,**
- **Phát hiện vật thể trong ảnh:** Xe, con người, đèn giao thông, etc.
- **Tạo ảnh với những phong cách khác nhau:** Hiện thị nội dung ngữ nghĩa của ảnh gốc theo những phong cách khác nhau.

Mạng Nơ-ron truyền thống (Neural Network) hoạt động không thực sự hiệu quả với dữ liệu đầu vào là hình ảnh. Nếu coi mỗi điểm ảnh là một thuộc tính (feature), một ảnh RGB kích thước (64×64) có 12288 ($= 64 \times 64 \times 3$) thuộc tính. Nếu kích thước ảnh tăng lên 1000×10000 , chúng ta có 3 triệu ($3M$) thuộc tính cho mỗi ảnh đầu vào. Nếu sử dụng mạng liên kết đầy đủ (*fully connected NN*) và giả sử lớp thứ 2 có 1000 thành phần (units/ neurons), ma trận trọng số sẽ có kích thước $1000 \times 3M$ tương đương với $3B$ trọng số cần huấn luyện (learning). Điều này yêu cầu khối lượng tính toán cực lớn (expensive computational cost) và thường dẫn đến overfitting (<https://en.wikipedia.org/wiki/Overfitting>) do không đủ dữ liệu huấn luyện.

2. Mạng nơ-ron tích chập (Convolutional Neural Network)


Mạng nơ-ron tích chập (CNN hay ConvNet) là mạng nơ-ron (Wikipedia (https://en.wikipedia.org/wiki/Artificial_neural_network), bài báo (<https://www.sciencedirect.com/science/article/pii/S0731708599002721>), video (<https://www.youtube.com/watch?v=aircAruvnKk>)) phổ biến nhất được dùng cho dữ liệu ảnh. Bên cạnh các lớp liên kết đầy đủ (FC layers), CNN còn đi cùng với các lớp ẩn đặc biệt giúp phát hiện và trích xuất những đặc trưng - chi tiết (patterns) xuất hiện trong ảnh gọi là **Lớp Tích chập (Convolutional Layers)**. Chính những lớp tích chập này làm CNN trở nên khác biệt so với mạng nơ-ron truyền thống và hoạt động cực kỳ hiệu quả trong bài toán phân tích ảnh.

Lớp tích chập (Convolutional Layers)

 Lớp tích chập được dùng để phát hiện và trích xuất đặc trưng - chi tiết của ảnh.

Giống như các lớp ẩn khác, lớp tích chập lấy dữ liệu đầu vào, thực hiện các phép chuyển đổi để tạo ra dữ liệu đầu vào cho lớp kế tiếp (đầu ra của lớp này là đầu vào của lớp sau). Phép biến đổi được sử dụng là phép tính tích chập. Mỗi lớp tích chập chứa một hoặc nhiều bộ lọc - bộ phát hiện đặc trưng (filter - feature detector) cho phép phát hiện và trích xuất những đặc trưng khác nhau của ảnh.

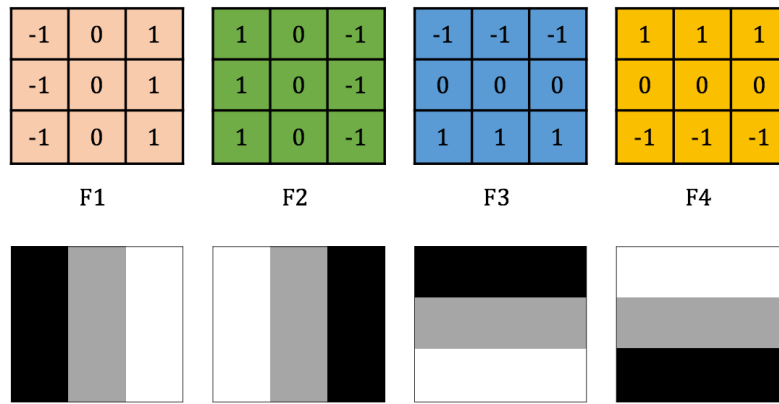
Đặc trưng của ảnh là gì? Đặc trưng ảnh là những chi tiết xuất hiện trong ảnh, từ đơn giản như cạnh, hình khối, chữ viết tới phức tạp như mắt, mặt, chó, mèo, bàn, ghế, xe, đèn giao thông, v.v.. Bộ lọc phát hiện đặc trưng là bộ lọc giúp phát hiện và trích xuất các đặc trưng của ảnh, có thể là bộ lọc góc, cạnh, đường chéo, hình tròn, hình vuông, v.v.

 Bộ lọc ở lớp tích chập càng sâu thì phát hiện các đặc trưng càng phức tạp.

Độ phức tạp của đặc trưng được phát hiện bởi bộ lọc tỉ lệ thuận với độ sâu của lớp tích chập mà nó thuộc về. Trong mạng CNN, những lớp tích chập đầu tiên sử dụng bộ lọc hình học (geometric filters) để phát hiện những đặc trưng đơn giản như cạnh ngang, dọc, chéo của bức ảnh. Những lớp tích chập sau đó được dùng để phát hiện đối tượng nhỏ, bán hoàn chỉnh như mắt, mũi, tóc, v.v. Những lớp tích chập sâu nhất dùng để phát hiện đối tượng hoàn chỉnh như: chó, mèo, chim, ô tô, đèn giao thông, v.v. Để hiểu cách thức hoạt động của lớp tích chập cũng như phép tính tích chập, hãy cùng xem ví dụ về bộ lọc phát hiện cạnh (edge filters/ detectors) dưới đây.

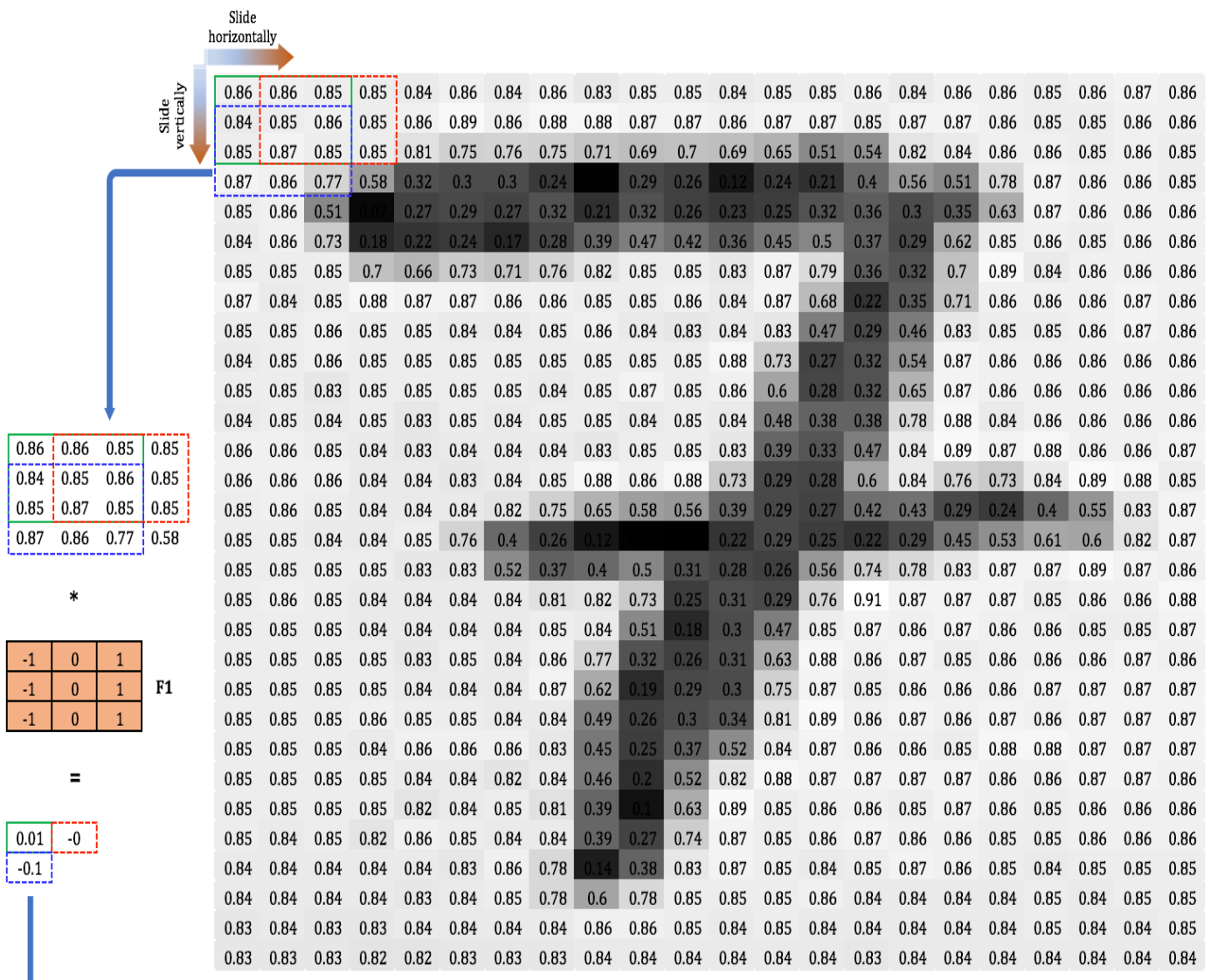
Ví dụ về bộ lọc cạnh

Trong ví dụ này, CNN được sử dụng để phân loại tập các ảnh viết tay của các số từ 0 tới 9. Đầu vào là những bức ảnh trắng đen (Gray Scale) và được biểu diễn bởi một ma trận các điểm ảnh với kích thước cố định $h \times w$. Lớp tích chập đầu tiên của CNN sử dụng 4 bộ lọc kích thước 3×3 : F_1, F_2, F_3, F_4 với giá trị tương ứng như trong hình 1. Các giá trị tại mỗi ô của các bộ lọc có thể được biểu diễn bởi màu sắc tương ứng với Đen (−1), Xám (0), Trắng (1) như trong hình dưới đây.



Hình 1: Bộ lọc được sử dụng trong lớp tích chập đầu tiên là các ma trận kích thước 3×3 của -1 , 0 và 1 .

Để minh họa cho phép nhân chập, chúng ta sử dụng đầu vào là một bức ảnh viết tay của số 7, biểu diễn dưới dạng ma trận 30×22 và áp dụng riêng biệt từng bộ lọc ở trên. Phép nhân tích chập được thực hiện bằng cách trượt ma trận lọc 3×3 trên ma trận ảnh đầu vào 32×22 (bộ lọc dịch sang phải/ xuống dưới 1 cột/ hàng mỗi một lần trượt) cho đến khi nó đi qua hết tất cả các vùng kích thước 3×3 . Việc trượt ma trận lọc trên ma trận đầu vào được gọi là “chập” (convolving). Như minh họa trong hình 2, ma trận F_1 được chập với từng vùng (block - region) điểm ảnh kích thước 3×3 của ảnh đầu vào. Tại mỗi vị trí di chuyển của ma trận F_1 , giá trị đầu ra được tính bằng tích chập (dot-product) của ma trận F_1 với vùng bao phủ tương ứng.



0.01	-0	-0	-0	-0.1	-0	-0	-0.1	0	-0	-0	-0.2	-0.1	0.3	0.32	0.05	-0	-0	0.03	0.01
-0.1	-0.3	-0.5	-0.3	-0.1	-0.1	-0.3	-0	0.19	-0.2	-0.1	-0.1	0.03	0.66	0.43	0.25	0.36	0.06	0	0
-0.4	-1.1	-0.7	-0.2	-0.1	-0	-0.4	-0	0.25	-0.3	-0.1	0	0.16	0.64	0.4	0.59	0.9	0.3	-0	-0
-0.6	-1.6	-1.2	0	-0.1	0.01	-0.1	0.24	0.29	-0.4	0	0.32	0.19	0.12	0.35	1.11	1.12	0.31	-0	0
-0.5	-1.6	-0.9	0.31	0	0.1	0.27	0.28	0.11	-0.2	0.04	0.19	-0.5	-0.7	0.58	1.46	0.9	0.2	0.01	0.01
-0.1	-0.8	-0.7	0.08	-0	0.06	0.32	0.27	0.07	-0.1	0.06	-0.1	-1.2	-1	1.08	1.64	0.53	-0	0.03	0.01
-0	-0.1	-0.2	0.01	0.03	0.03	0.12	0.07	0.01	-0	0.03	-0.6	-1.7	-0.8	1.37	1.47	0.31	-0	0.05	0
0.01	0.04	0	-0	-0	0	0.01	-0	-0	0.02	-0.1	-1.1	-1.6	-0.1	1.58	1.22	0.16	0.01	0.03	0
0.01	0	0	-0	-0	0	0.02	0.02	-0	0.02	-0.4	-1.6	-1.2	0.63	1.64	0.92	0	0.01	0.02	0
0	0	0	0	0.01	-0	0.01	0.02	0	0.02	-0.7	-1.7	-0.8	1.04	1.6	0.59	-0	0.02	0	0
-0	-0	-0	0	0.02	-0	0	0.03	0.02	-0	-1.1	-1.5	-0.3	1.28	1.47	0.3	-0	0.01	-0	0.01
-0	-0	-0.1	-0	0.02	0.02	0.04	0.01	0.02	-0.2	-1.4	-1.4	0.29	1.47	1.08	-0	0.05	0.17	0.02	-0
-0	-0.1	-0.1	-0	-0	-0.1	-0.1	-0.2	-0.1	-0.3	-1.3	-1.1	0.52	1.23	0.45	-0.3	0.18	0.46	0.45	0.29
-0	-0.1	-0	-0.1	-0.5	-0.6	-0.4	-0.4	-0.2	-0.2	-0.6	-0.5	0.37	0.76	0.26	-0.1	0.35	0.54	0.68	0.55
-0	-0	-0	-0.1	-0.8	-1.1	-0.6	-0.3	-0.3	-0.2	-0.1	0.19	0.54	0.42	0.19	0.14	0.31	0.4	0.64	0.56
-0	-0	-0	-0.1	-0.8	-1	-0.4	-0.2	-0.7	-0.5	0.24	0.76	1.03	0.37	0.28	0.33	0.18	0.08	0.22	0.26
0	-0	-0	-0	-0.3	-0.5	-0.1	-0.3	-1.3	-0.9	0.28	1.28	1.5	0.34	0.05	0.09	0.01	0	0	0.01
0	-0	-0	0	0.01	-0	-0.1	-1	-1.7	-0.6	0.7	1.57	1.25	0.11	-0.1	-0	-0	-0	0.01	0.04
0	-0	-0	-0	0.01	0.05	-0.3	-1.6	-1.5	-0.1	1.12	1.69	0.73	-0	0	-0	0.01	0	0	0.02
0	0.01	-0	-0	0	0.03	-0.6	-1.6	-1	0.18	1.34	1.69	0.38	-0	0	-0	0.02	0.01	0.02	0
0	0	0	0	-0	-0	-1	-1.6	-0.6	0.46	1.44	1.47	0.17	-0	0	0.02	0.04	0	0	0
0	0	0	0	-0	-0	-1.1	-1.6	-0.2	0.97	1.34	0.95	0.06	-0	-0	0.01	0.02	0	0.01	-0
0	-0	-0	0	0.01	-0.1	-1.2	-1.6	0.22	1.68	1.05	0.37	0.02	-0	0	0.02	0	0	0.01	-0
0	-0	-0	0.01	-0	-0	-1.3	-1.6	0.65	2.01	0.69	0.01	0.02	-0	0	-0	-0	0.02	0.03	-0
0	-0	-0	0.01	0.03	-0.1	-1.6	-1.7	1.28	1.88	0.35	-0.1	0.03	0.02	0.01	-0	-0.1	0.01	0.03	0
0	-0	0	0.02	0.02	-0.1	-1.4	-1	1.29	1.16	0.13	-0	0.01	0.01	0	-0	-0	0.01	0.02	0.01
0	-0	0	0	0.04	-0.1	-1	-0.4	0.93	0.54	0.02	-0	-0	0.01	0.01	-0	0	0	0	0.02
0	-0	-0	0.02	0.03	-0.1	-0.2	0.03	0.24	0.05	0	0.01	-0	-0	0.01	0	0.02	0	-0	0.02

Hình 2: Nhân chập bộ lọc F_1 với ma trận ảnh đầu vào của số 7

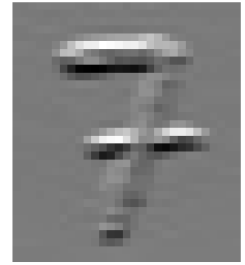
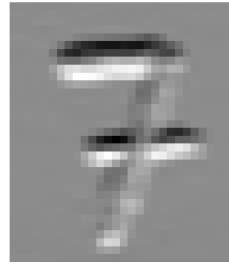
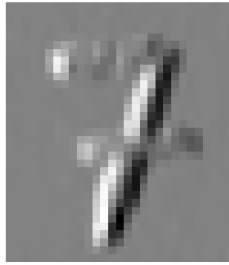
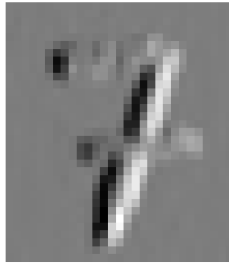
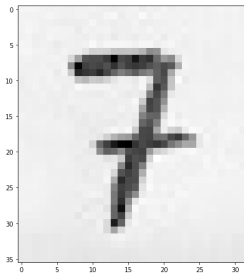
Ô đầu tiên (0, 0) của ma trận đầu ra (giá trị 0.01) ra là kết quả của phép nhân chập giữa ma trận F_1 với góc trái trên cùng của ma trận đầu vào và được tính như sau:

$$\begin{bmatrix} 0.86 & 0.86 & 0.85 \\ 0.84 & 0.85 & 0.86 \\ 0.85 & 0.87 & 0.85 \end{bmatrix} * \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.86*(-1) + 0.86*0 + 0.85*1 \\ 0.84*(-1) + 0.85*0 + 0.86*1 \\ 0.85*(-1) + 0.87*0 + 0.85*1 \end{bmatrix} = \begin{bmatrix} 0.01 \\ 0.01 \\ 0.01 \end{bmatrix}$$

3x3 block Dot product (Element-wise) F_1

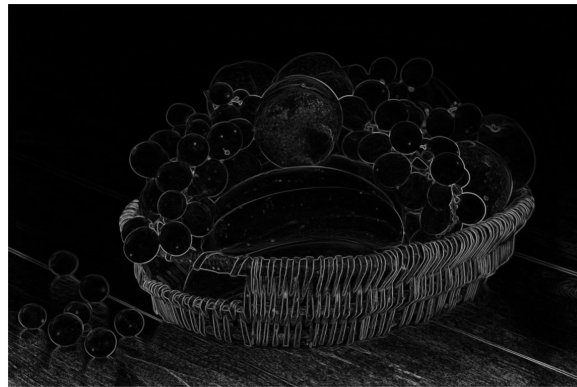
Từ các ma trận đầu ra kích thước 28×20 , chúng ta thấy được cả bốn bộ lọc F_1 , F_2 , F_3 và F_4 đều được sử dụng để phát hiện cạnh trong bức ảnh (thể hiện bởi những điểm ảnh sáng hơn) (Hình 3):

- F_1 : Phát hiện cạnh đứng phải.
- F_2 : Phát hiện cạnh đứng trái.
- F_3 : Phát hiện cạnh ngang dưới.
- F_4 : Phát hiện cạnh ngang trên.



Hình 3: Ví dụ về bộ lọc cạnh (đứng phải, đứng trái, ngang dưới, ngang trên) với đầu vào là ảnh số viết tay.

Các bộ lọc cạnh: Rất nhiều bộ lọc cạnh đã được đề xuất với sự khác biệt nhỏ về kích thước và giá trị. Các ma trận lọc này thường có kích thước 3×3 với các giá trị nhỏ trong khoảng -5 tới 5 và đối xứng. Trong hình 4, bộ lọc Sobel (https://en.wikipedia.org/wiki/Sobel_operator) được sử dụng để phát hiện các cạnh của giỏ hoa quả (ảnh màu) và cho kết quả khá tốt.



Hình 4: Ví dụ về bộ lọc cạnh với ma trận lọc Sobel.

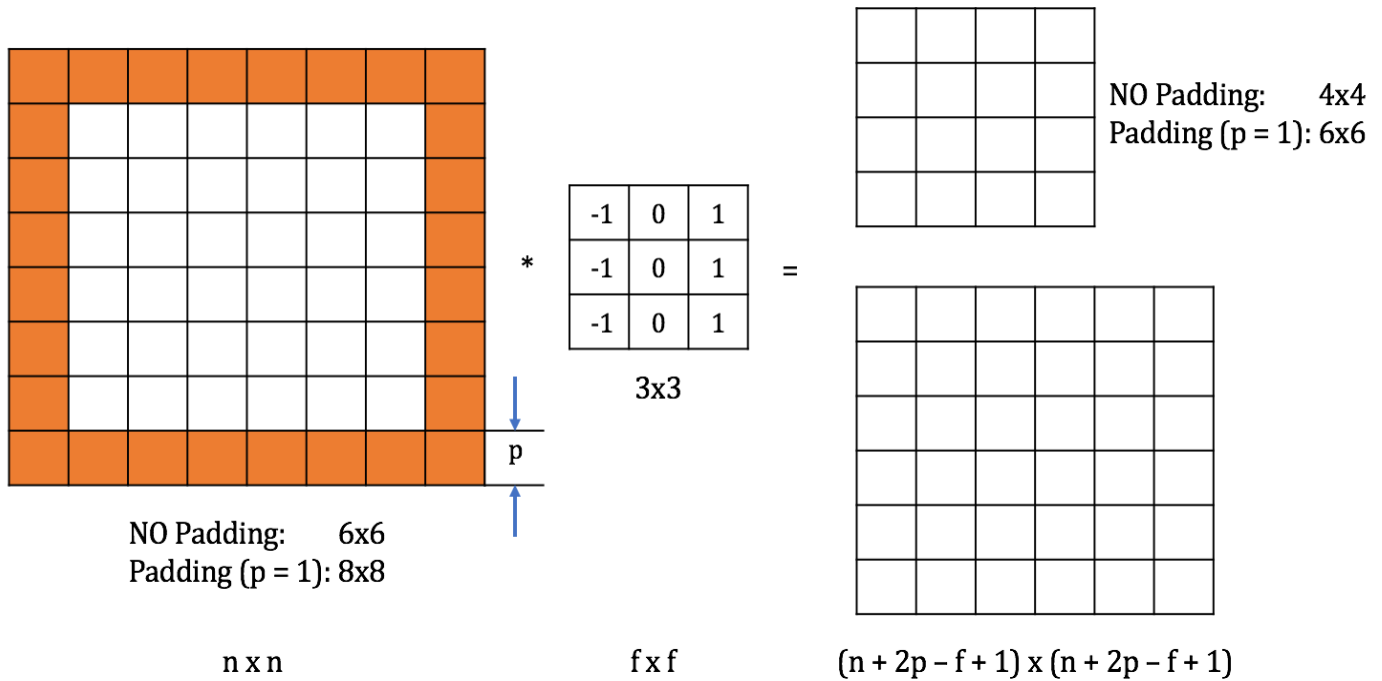
Nếu ta có tập dữ liệu huấn luyện lớn và hiệu năng tính toán cao, với những tập ảnh kích thước lớn và nhiều chi tiết phức tạp, các bộ lọc cạnh có thể được huấn luyện tự động từ tập dữ liệu. Nghĩa là các giá trị của ma trận lọc được coi như tham số của một mạng nơ-ron và huấn luyện (sử dụng back-propagation chẳng hạn) để có một tập giá trị tối ưu. Với cách tiếp cận này, các bộ lọc tạo ra có thể phát hiện không chỉ cạnh đứng hay ngang mà còn có thể những cạnh nghiêng một góc lẻ như 40° , 45° hoặc 70° .

Padding

Ảnh hưởng của phép tích chập

- Lấy ví dụ với ma trận đầu vào kích thước 6×6 . Nếu ta nhân chập với bộ lọc kích thước 3×3 , kết quả thu được là một ma trận đầu ra kích thước 4×4 vì chỉ có 4×4 vị trí trên ma trận đầu vào để đặt ma trận lọc. Tổng quát hoá, nếu ta nhân chập ma trận đầu vào kích thước $n \times n$ với bộ lọc kích thước $f \times f$, ta thu được kết quả là một ma trận kích thước $(n - f + 1) \times (n - f + 1)$. Mỗi một lần áp dụng phép nhân chập, kích thước của ảnh bị giảm xuống, và vì thế chúng ta chỉ có thể thực hiện nó một vài lần trước khi ảnh trở nên quá nhỏ.
- Điểm ảnh ở khoảng trung tâm của ma trận đầu vào được bao phủ bởi rất nhiều vùng 3×3 nghĩa là được sử dụng để tính nhiều giá trị đầu ra, trong khi những điểm ảnh ở góc hoặc cạnh chỉ được sử dụng 1 hoặc 2 lần vì chỉ bị bao phủ bởi 1 hoặc 2 vùng 3×3 . Vì

thế chúng ta đánh mất rất nhiều thông tin (có thể quan trọng) tại các vùng gần cạnh của ảnh.



Hình 5: Ma trận đầu vào được bao quanh bởi đường viền phụ kích thước p (giá trị 0).

Để khắc phục hai nhược điểm trên, một đường viền phụ (padding) được thêm vào xung quanh ma trận đầu vào. Việc thêm đường viền phụ làm tăng kích thước của ma trận đầu vào, dẫn tới tăng kích thước ma trận đầu ra. Từ đó độ chênh lệch giữa ma trận đầu ra với ma trận đầu vào gốc giảm. Những ô nằm trên cạnh/ góc của ma trận đầu vào gốc cũng lùi sâu vào bên trong hơn, dẫn tới được sử dụng nhiều hơn trong việc tính toán ma trận đầu ra, tránh được việc mất mát thông tin.

Trong hình 5, ma trận đầu vào kích thước 6×6 được thêm vào đường viền phụ kích thước 1 ($p = 1$), trở thành ma trận 8×8 . Khi nhân chập ma trận này với bộ lọc 3×3 , chúng ta thu được ma trận đầu ra 6×6 . Kích thước của ma trận đầu vào (gốc) được duy trì. Những điểm ảnh nằm ở cạnh của ma trận đầu vào gốc được sử dụng nhiều lần hơn (4 lần với những điểm ảnh ở góc).

Theo quy ước, các ô trên đường viền phụ có giá trị bằng không, p là kích thước của đường viền phụ. Trong hầu hết các trường hợp, đường viền phụ đối xứng trái-phải, trên-dưới so với ma trận gốc, vì thế kích thước của ma trận đầu vào được tăng lên $2p$ mỗi chiều. Ma trận đầu ra do đó có kích thước $(n + 2p - f + 1) \times (n + 2p - f + 1)$.

Tùy theo giá trị của p , chúng ta có hai trường hợp chính:

- Nhân chập không dùng đường viền phụ (**valid convolution**) - NO padding:

$$(n \times n) * (f \times f) \Rightarrow (n - f + 1) \times (n - f + 1)$$

- Nhân chập không làm thay đổi kích thước đầu vào (**same convolution**): Kích thước đường viền phụ được tính theo công thức:

$$n + 2p - f + 1 = n \Rightarrow p = \frac{f - 1}{2}$$

Theo quy ước, kích thước bộ lọc f là số lẻ vì hai lý do chính sau:

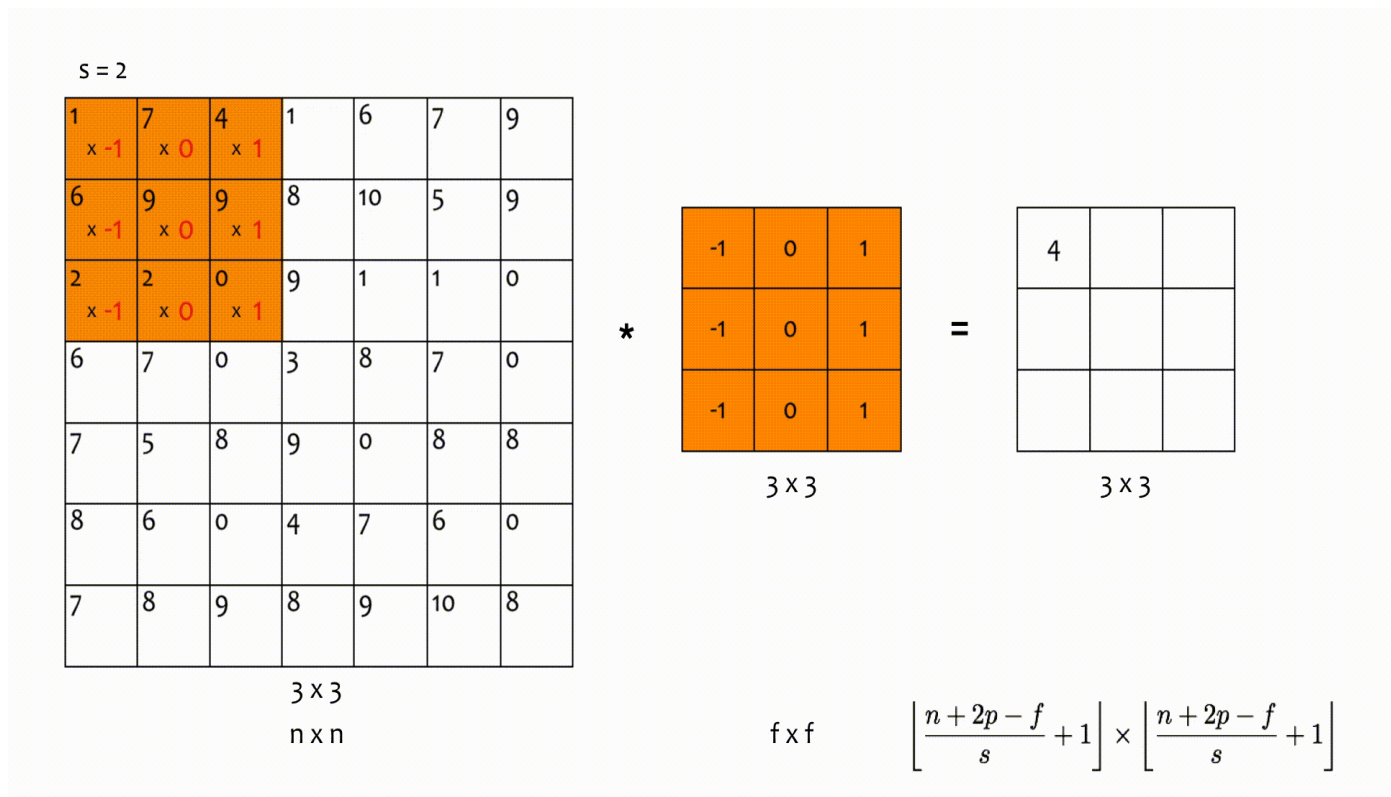
- Nếu f là số chẵn, chúng ta phải thêm vào bên trái của ma trận đầu vào nhiều hơn bên phải (hoặc ngược lại), việc này dẫn tới hệ đầu vào không đối xứng (**asymetric**).
- Nếu f là số lẻ, ma trận đầu vào có một điểm ảnh ở trung tâm. Trong lĩnh vực thị giác máy tính, việc có một nhân tố khác biệt (distinguisher) - một điểm đại diện cho vị trí của bộ lọc thường mang lại hiệu năng cao cho bài toán.

Nhân chập sai (strided convolutions)

Trong phép nhân chập ở trên, bộ lọc trượt trên ma trận đầu vào 1 hàng/ cột trong mỗi bước di chuyển. Tuy nhiên, giá trị này có thể bằng 2, 3 hoặc lớn hơn. Số hàng/ cột mà bộ lọc trượt qua trong một bước di chuyển ký hiệu là s . Kích thước ma trận đầu ra lúc này được tính bởi:

$$\left(\frac{n + 2p - f}{s} + 1 \right) \times \left(\frac{n + 2p - f}{s} + 1 \right)$$

Nếu $n + 2p - f$ không chia hết cho s , chúng ta lấy chận dưới ($\lfloor \rfloor$) như trong hình minh hoạ dưới đây.



Hình 6: Nhân chập với bước sai (trượt) $s = 2$.

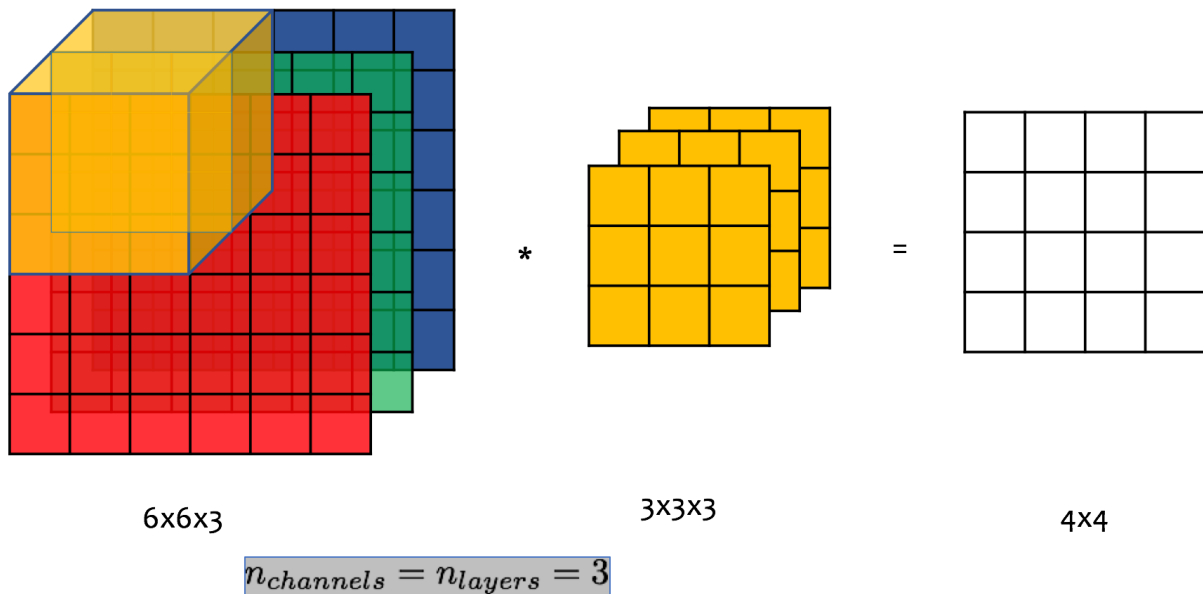
Trong lĩnh vực toán học thuần túy, phép toán nhân chập được định nghĩa hơi khác so với phía trên. Trước khi thực hiện nhân chập (element-wise/ dot-product) và lấy tổng của các kết quả thu được, bộ lọc (filter) được lật lần lượt theo trục ngang và trục dọc (**flipped filter**). Ma trận đầu ra được tính dựa trên ma trận đầu vào và bộ lọc đã được lật này. Phép toán “nhân chập” được trình bày ở trên (thực hiện trực tiếp trên ma trận đầu vào và bộ lọc gốc) được gọi là tương quan chéo (**cross-correlation**). Tuy nhiên, theo quy ước trong ML và DL, phép tương quan chéo (cross-correlation) được gọi là phép nhân chập (convolution).

3. Phép chập khối

Phép chập khối với một bộ lọc

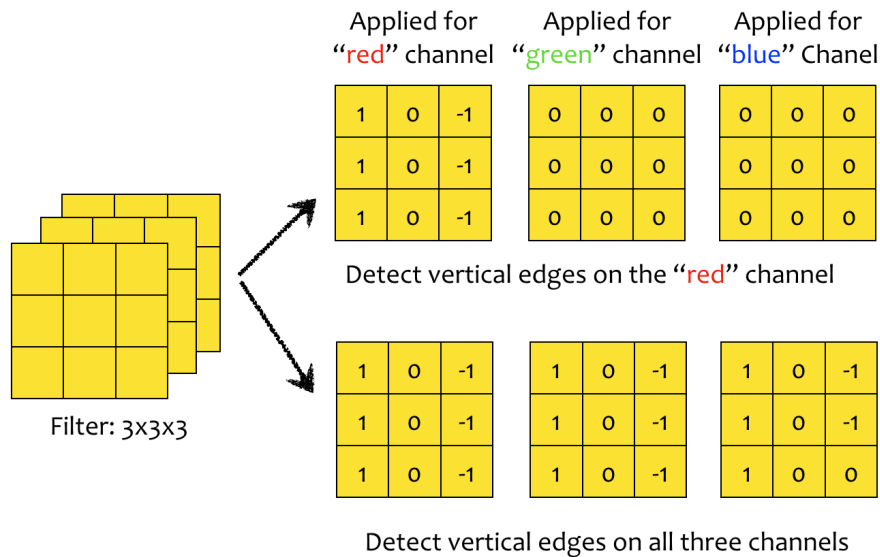
 Số lớp (layers) của bộ lọc phải bằng số kênh (channels) của ảnh đầu vào.

Các ví dụ ở trên sử dụng ảnh trong hệ gray và được biểu diễn dưới dạng ma trận 2 chiều (2D). Phép nhân chập cũng có thể dùng cho ảnh màu (3D images). Giả sử chúng ta có ảnh đầu vào kích thước 6×6 được biểu diễn trong hệ RGB. Ma trận đầu vào do đó có kích thước $6 \times 6 \times 3$ (3 kênh màu). Bộ lọc được sử dụng do đó cũng phải có 3 lớp tương ứng với 3 kênh màu **đỏ**, **xanh lục** và **xanh lam**.



Hình 7: Phép nhân chập khối - áp dụng cho ảnh màu RGB kích thước **6x6**. Khối lọc (filter cube) được dịch chuyển trên khối ma trận đầu vào. Mỗi lớp của bộ lọc được nhân chập với phần diện tích bị bao phủ bởi nó trên kênh tương ứng của ma trận đầu vào. Tại một vị trí cụ thể của khối lọc, giá trị tại ô tương ứng của trận đầu ra (ma trận 2 chiều) là tổng của ba tích thu được.

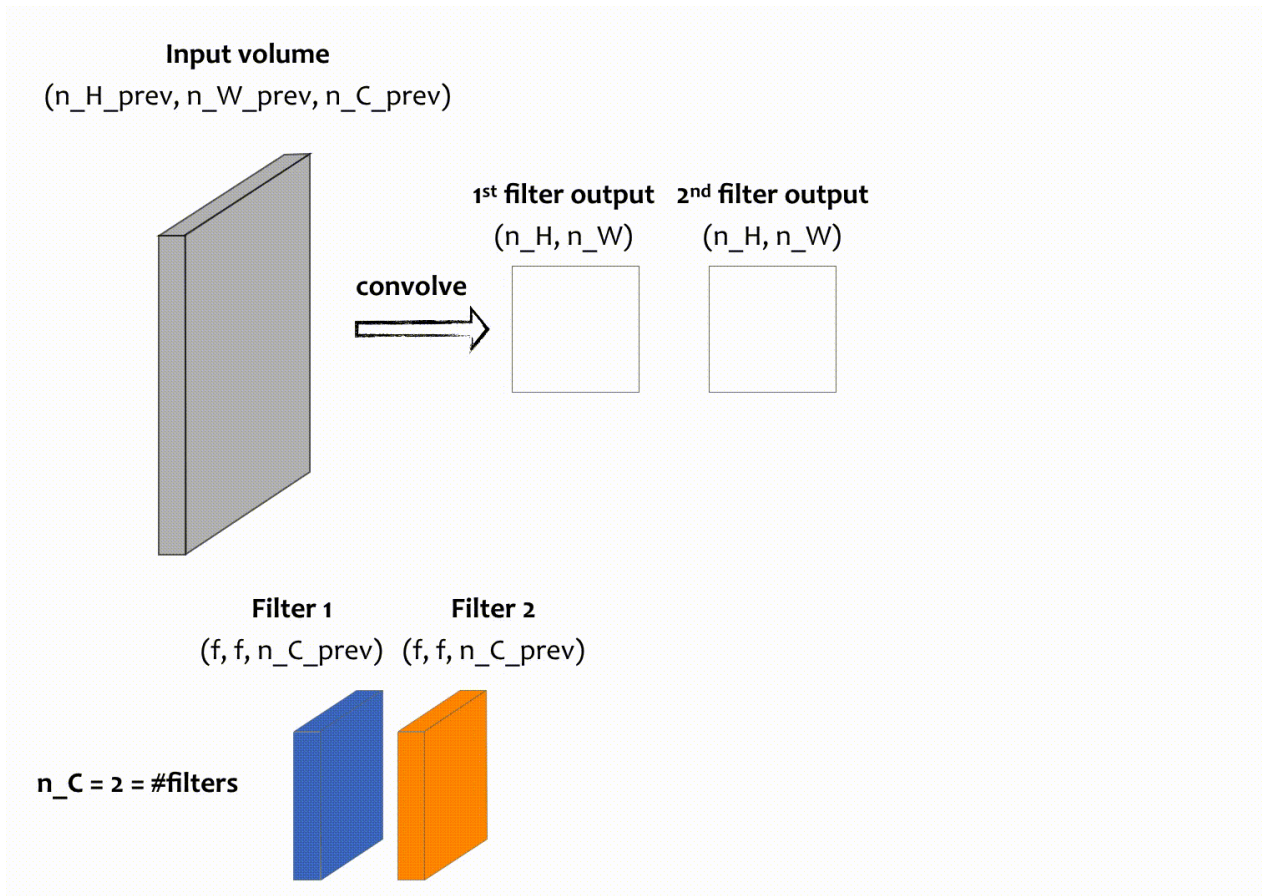
Phép chập khối có thể được sử dụng để phát hiện và trích xuất đặc trưng (chi tiết) ảnh trên từ kênh màu. Ví dụ, để phát hiện cạnh trên kênh màu **đỏ** (red), chúng ta đặt bộ phát hiện cạnh ở lớp đầu tiên của bộ lọc và thiết lập hai lớp kế tiếp bằng 0 (không thực hiện gì cả). Tương tự, để phát hiện cạnh trên cả ba kênh màu, bộ phát hiện cạnh được đặt ở cả ba lớp của bộ lọc khối (Hình 8).



Hình 8: Ba lớp của bộ lọc có thể được cấu hình khác nhau để phát hiện đặc trưng trên một, hai hoặc cả ba kênh màu của ảnh đầu vào.

Phép chập khối với nhiều bộ lọc

Tại một lớp tích chập, nhiều bộ lọc có thể được sử dụng cùng lúc để phát hiện những đặc trưng khác của ảnh ví dụ như cạnh đứng, ngang hay nghiêng 45° . Trong hình 8 hai bộ lọc kích thước $3 \times 3 \times 3$ được sử dụng cùng lúc để đồng thời phát hiện cạnh đứng và ngang. Với mỗi bộ lọc, ta thu được ma trận có kích thước 4×4 như đã trình bày ở trên. Hai ma trận này được nhập lại (stack together) tạo thành một ma trận đầu ra duy nhất kích thước $4 \times 4 \times 2$ (Hình 9).

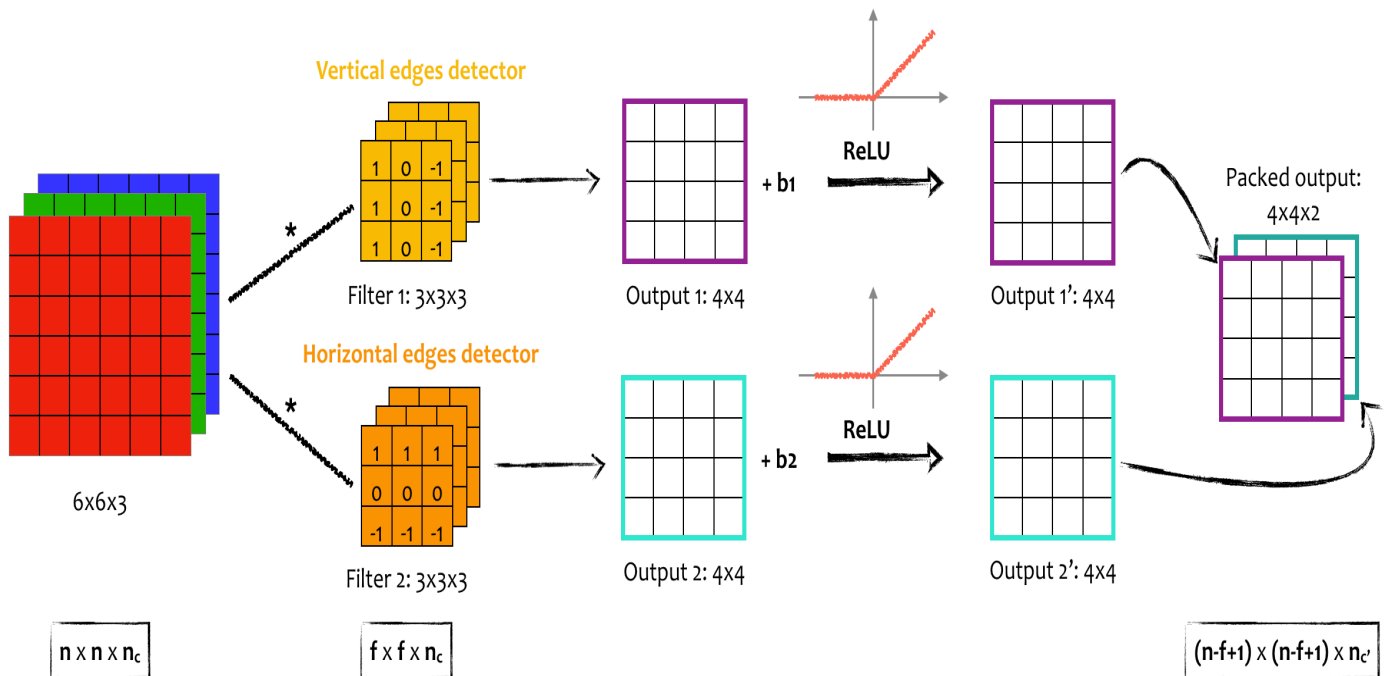


Hình 9: Hai bộ lọc kích thước $3 \times 3 \times 3$ được sử dụng để phát hiện đồng thời cạnh đứng và ngang của ảnh đầu vào (hệ RGB).

4. Mạng CNN một lớp

Kiến trúc

Phép chập với nhiều bộ lọc trình bày ở trên có thể chuyển thành CNN một lớp bằng cách cộng thêm vào mỗi ma trận ra 4×4 một số thực b (bias) và đưa chúng qua một hàm kích hoạt không tuyến tính (non-linear activation function), ví dụ như *ReLU*. Kết hợp hai ma trận thu được, ta được khối ma trận ra kích thước $4 \times 4 \times 2$.



Hình 10: Kiến trúc của một lớp: **Input** \Rightarrow 2 filters of $3 \times 3 \times 3 \Rightarrow$ **ReLU** (non-linear activation function) \Rightarrow **Output**.

Tham số

Có bao nhiêu tham số trong một lớp tích chập sử dụng 10 bộ lọc kích thước $3 \times 3 \times 3$? **Trả lời:** Mỗi bộ lọc kích thước $3 \times 3 \times 3$ có 27 tham số, cộng thêm bias b . Vì thế, có tất $28 \times 10 = 280$ tham số cho 10 bộ lọc. Lưu ý rằng số lượng tham số là cố định và hoàn toàn không phụ thuộc vào kích thước của ảnh đầu vào (1000×1000 hay 5000×5000). Đây là một tính chất quan trọng của lớp tích chập giúp CNN tránh được rủi ro overfitting do có quá nhiều tham số nếu sử dụng lớp liên kết đầy đủ (như đã trình bày ở trên).

Ký hiệu

If layer l is a convolution layer, so: **Size**

- $f^{[l]}$ = filter size;
- $p^{[l]}$ = padding;
- $s^{[l]}$ = stride;
- $n_c^{[l]}$ = number of filters
- Input: $n_H^{[l-1]} \times n_W^{[l-1]} \times n_c^{[l-1]}$
- Output: $n_H^{[l]} \times n_W^{[l]} \times n_c^{[l]}$
- Each filter: $f^{[l]} \times f^{[l]} \times n_c^{[l-1]}$ (#filter's layers = #input channels)
- Weights: $f^{[l]} \times f^{[l]} \times n_c^{[l-1]} \times n_c^{[l]}$ ($n_c^{[l]}$ is the number of filters in layer l)
- Activations: $a^{[l]} \Rightarrow n_H^{[l]} \times n_W^{[l]} \times n_c^{[l]}$; $A^{[l]} \Rightarrow m \times n_H^{[l]} \times n_W^{[l]} \times n_c^{[l]}$ (m is number of training examples)
- Bias: $n_c^{[l]} - (1, 1, 1, n_c^{[l]})$

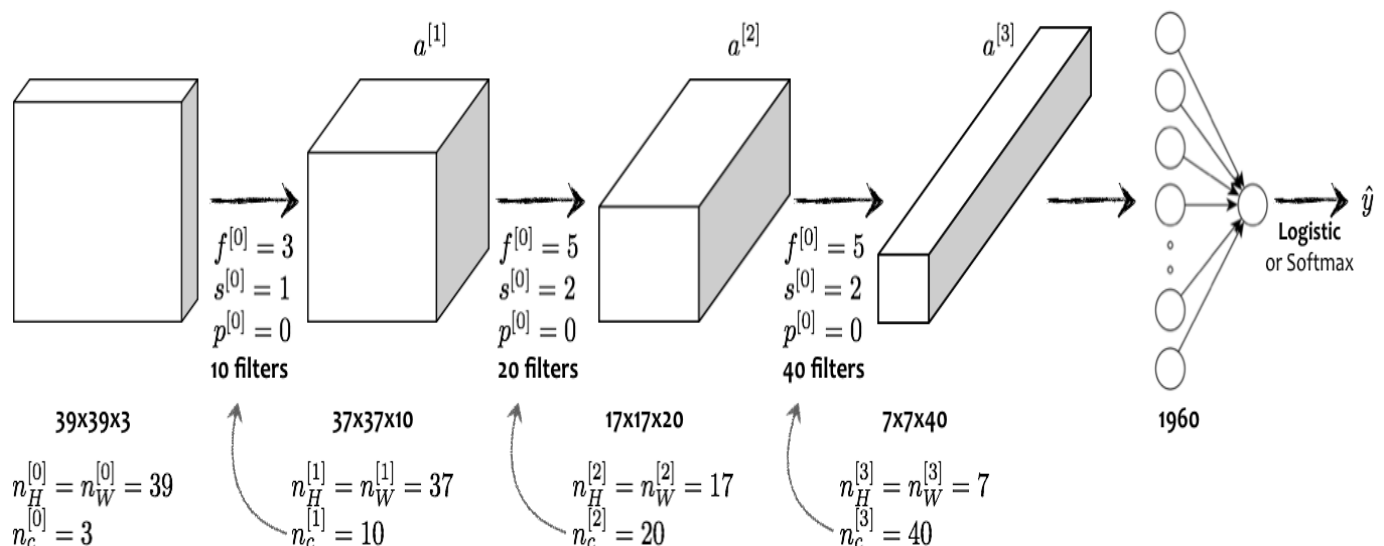
#filter's layers = #input channels

Hình 11: Các ký hiệu cơ bản của một CNN.

5. CNN đơn giản

Hãy lấy ví dụ về một CNN nhiều lớp (deep CNN) lấy đầu vào là ảnh X và xác định xem nó có phải là ảnh **mèo** hay không (bài toán phân loại ảnh - **image classification**). CNN được thiết lập như sau:

- **Ảnh đầu vào** có kích thước $39 \times 39 \times 3$ (hệ RGB).
- Lớp tích chập **đầu tiên** sử dụng 10 bộ lọc, $f^{[1]} = 5$, bước trượt $s^{[1]} = 1$, không padding $p^{[1]} = 0$.
- Lớp tích chập **thứ hai** sử dụng 20 bộ lọc, $f^{[2]} = 5$, bước trượt $s^{[2]} = 2$, không padding $p^{[2]} = 0$.
- Lớp tích chập **cuối cùng** sử dụng 40 bộ lọc, $f^{[3]} = 5$, bước trượt $s^{[3]} = 2$, không padding $p^{[3]} = 0$.
- **Phẳng hoá** (flatten - unroll) ma trận khối thu được thành một vector cột chứa 1960 phần tử.
- Sử dụng lớp **logistic regression** (https://en.wikipedia.org/wiki/Logistic_regression) để thu về **kết quả**: 0 (không phải mèo), 1 (mèo).



Hình 12: Ví dụ một CNN cơ bản được sử dụng cho bài toán phân loại ảnh (“mèo” hay “không phải mèo”).

Hình 12 minh họa một ví dụ cơ bản của CNN. Cấu trúc của các CNNs khá tương đồng nên việc lựa chọn các siêu thông số sử dụng trong các CNNs thường được chú trọng hơn: kích thước của bộ lọc, giá trị của bước trượt (s), độ rộng đường viền phụ padding (p), số lượng bộ lọc dùng trong một lớp chập (n_c).

 **Càng về cuối của CNN, kích thước của ảnh càng giảm xuống trong khi số chiều thì tăng dần.**

Ba lớp cơ bản được sử dụng trong một CNN:

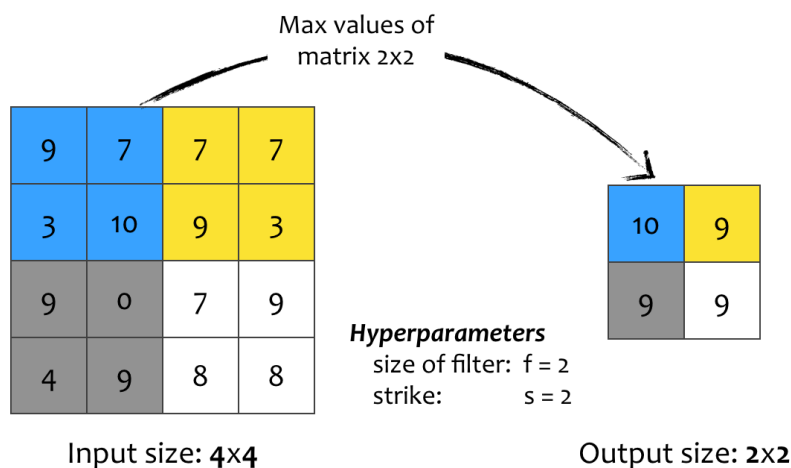
- Lớp tích chập: Convolution (CONV)
- Lớp Pooling: Pooling layer (POOL)
- Lớp liên kết đầy đủ: Fully connected (FC)

Lớp Pooling

Lớp Pooling được sử dụng trong CNN để giảm kích thước đầu vào, tăng tốc độ tính toán và hiệu năng trong việc phát hiện các đặc trưng. Có nhiều hướng Pooling được sử dụng, trong đó phổ biến nhất là pooling theo giá trị cực đại (max pooling) và pooling theo giá trị trung bình (average pooling).

Pooling theo giá trị cực đại (Max pooling)

Nếu một đặc tính được phát hiện ở một vùng nào đó bị bao phủ bởi bộ lọc, giá trị cao nhất trong vùng sẽ được giữ lại tuy nhiên chưa ai giải thích được tại sao cách tiếp cận này lại hoạt động tốt trong thực nghiệm.



Hình 13: Ví dụ **pooling theo giá trị cực đại**. Bộ lọc kích thước **2x2** trượt trên ma trận đầu vào 2 hàng/cột trong mỗi bước nhảy ($s = 2$) và chia nó thành những vùng khác nhau. Mỗi ô trong ma trận đầu ra lấy giá trị lớn nhất của vùng tương ứng.

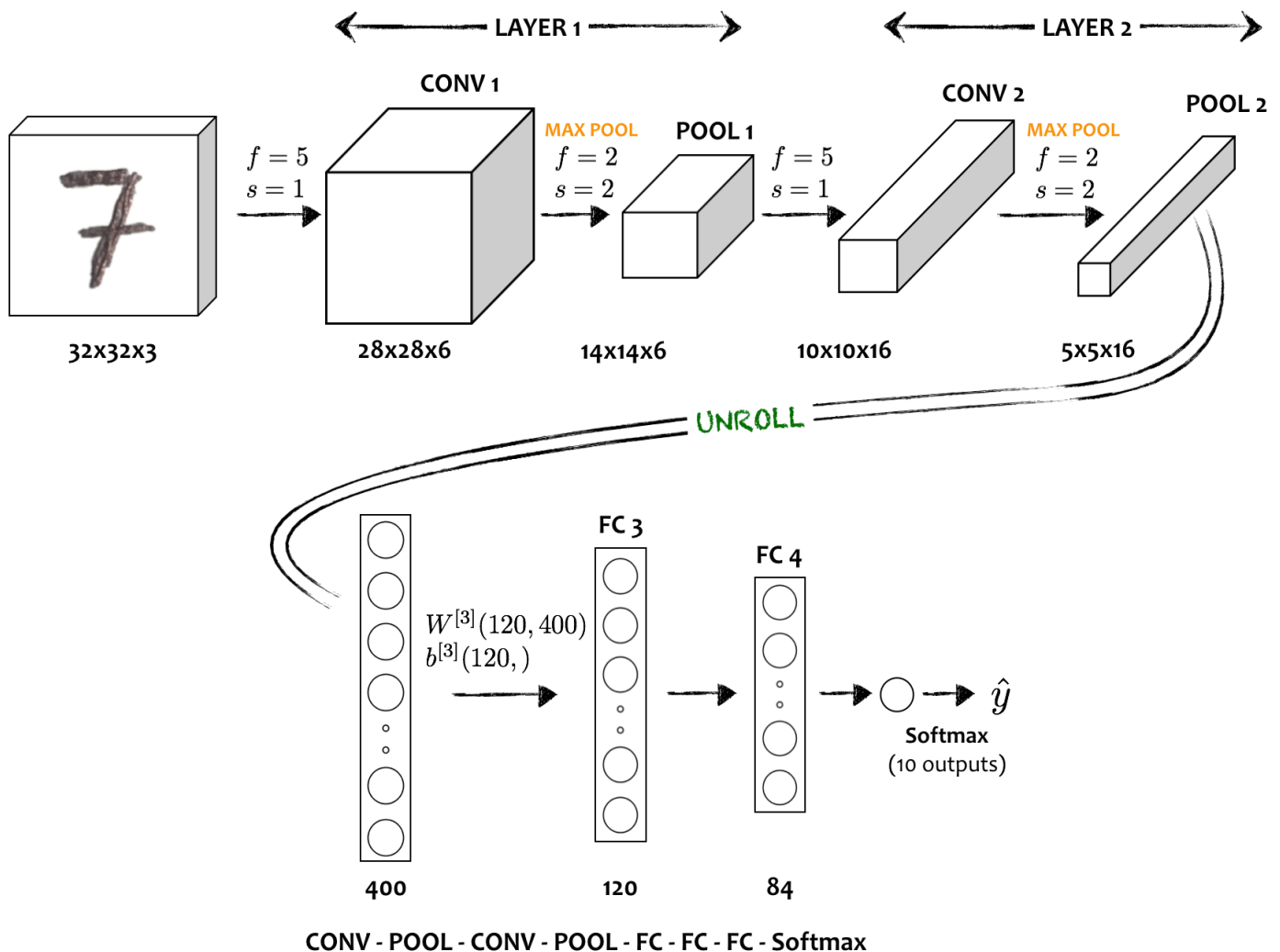
- Có hai siêu tham số **hyperparameters** (kích thước của bộ lọc f và giá trị bước sai s) tuy nhiên không có tham số cần huấn luyện trong lớp Max Pooling.
- Công thức tính kích thước ma trận ra không đổi: $\left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor$
- Với đầu vào là ma trận khối, việc tính toán trên các kênh được thực hiện độc lập.

Pooling theo giá trị trung bình (Average Pooling)

Thay vì lấy giá trị cực đại, pooling theo giá trị trung bình lấy trung bình của tất cả các giá trị trong vùng bị bao phủ bởi bộ lọc khi nó trượt trên ma trận đầu vào. Tuy nhiên pooling theo giá trị trung bình rất ít khi được sử dụng, hầu hết các CNNs hiện nay sử dụng pooling theo giá trị cực đại.

6. Ví dụ một CNN cụ thể

Một CNN đầy đủ thường chứa đồng thời lớp tích chập, lớp pooling, và lớp liên kết đầy đủ nằm liên tiếp nhau. Trong hình 14, CNN được sử dụng để phân phát hiện số viết tay trong ảnh (ví dụ với ảnh số 7). Như ta đã biết, càng về cuối của CNN, kích thước ảnh (n_H , n_W) giảm và số kênh (n_c) tăng. Trong CNN cơ bản, một hoặc hai lớp tích chập theo sau bởi lớp pooling được gộp chung thành một cụm và đôi khi được gọi là một lớp (lớp chứa lớp). Trong hình dưới đây, CNN có hai cụm (Layer 1 và Layer 2), mỗi cụm chứa một lớp tích chập và một lớp max pooling. Hai lớp liên kết đầy đủ (FC layers) ở cuối cùng, theo sau bởi một lớp Softmax (https://en.wikipedia.org/wiki/Softmax_function).



Hình 14: Ví dụ về một CNN đầy đủ dùng cho bài toán phân loại số viết tay. Cấu trúc cơ bản của một CNN thường là một vài cụm CONV => POOL theo sau bởi một tập FC và kết thúc bởi một lớp Softmax.


	Activation shape	Activation size	# parameters
Input	(32,32,3)	3072	0
CONV1 (f=5, s=1)	(28,28,8)	6272	208
POOL1	(14,14,8)	1568	0
CONV1 (f=5, s=1)	(10,10,16)	1600	416
POOL2	(5,5,16)	400	0
FC3	(120,1)	120	48001
FC4	(84,1)	84	10081
Softmax	(10,1)	10	841

Bảng 1: Tổng kết số lượng tham số tại mỗi lớp của CNN.

Kích thước hàm kích hoạt (activation shape/ size) và số lượng tham số tại từng lớp được thể hiện trong bảng 1. Chúng ta có thể nhận thấy:


- Lớp pooling (POOL) không có tham số.
- Số lượng tham số trong các lớp tích chập (CONV) không cao.
- Đa phần (rất nhiều) tham số tập trung ở các lớp liên kết đầy đủ (FC layers).
- Từ trái sang phải, kích thước các hàm kích hoạt có xu hướng giảm. Cần chú ý thiết lập giá trị các siêu tham số (p , s , f) vì kích thước các hàm kích hoạt giảm quá nhanh sẽ ảnh hưởng tiêu cực tới hiệu năng của cả CNN.

Tại sao lại tích chập?

 Hai ưu điểm chính của lớp tích chập khi so sánh với lớp liên kết đầy đủ là: Chia sẻ tham số và Liên kết thưa.

Nếu chúng ta có ảnh đầu vào kích thước $32 \times 32 \times 3$ và sử dụng 6 bộ lọc kích thước 5×5 , thì thu được đầu ra kích thước $28 \times 28 \times 6$. Đầu vào có 3072 ($= 32 \times 32 \times 3$) và đầu ra có 4704 ($= 28 \times 28 \times 6$) thành phần. Nếu sử dụng lớp liên kết đầy đủ, ma trận trọng số có kích thước 3072×4704 tương đương với gần 14M tham số. Trong khi đó, lớp tích chập chỉ có $6 \times (25 + 1) = 156$ tham số. Chúng ta có thể lý giải điều này theo hai cách sau:

- **Chia sẻ tham số:** Một bộ phát hiện đặc trưng (feature detector) ví dụ như bộ phát hiện cạnh hoạt động tốt trên một vùng của ảnh đầu vào thì cũng có thể hoạt động tốt trên các vùng còn lại. Các vùng bộ lọc đi qua trên ma trận đầu vào không tách biệt hoàn toàn, mà chia sẻ ít nhiều một phần diện tích (phụ thuộc vào bước trượt s). Điều này dẫn tới các tham số được dùng chung cho các vùng khác nhau cùng chứa nó trong việc tính toán giá trị đầu ra, từ đây số lượng tham số được giảm xuống. Điều này là hợp lý bởi nếu một tập tham số được có thể phát hiện cạnh ở góc trên bên trái của ảnh đầu vào, thì chúng cũng có thể được dùng để phát hiện cạnh ở góc phải bên dưới của ảnh, vì thế không nhất thiết phải sử dụng hai bộ phát hiện cạnh khác nhau cho hai vùng khác nhau của bức ảnh.
- **Liên kết thưa:** Một thành phần đầu ra (output unit) chỉ phụ thuộc vào bộ phát hiện đặc trưng và một phần nhỏ của ảnh đầu vào thay vì toàn bộ bức ảnh. Điều này khác với lớp liên kết đầy đủ, khi mỗi thành phần đầu ra phụ thuộc vào tất cả các thành phần của đầu vào.

 Một bức ảnh tạo thành bởi việc dịch chuyển một vài pixels sẽ có chung các đặc trưng với ảnh gốc và vì thế nên được gán cùng nhãn với ảnh gốc.

Huấn luyện tham số như thế nào?

Quá trình huấn luyện tham số không được đề cập chi tiết trong bài viết này, về cơ bản tham số được lựa chọn qua hai bước chính:

- Tính hàm giá trị (J) - thường là hàm mất mát dựa trên những tham số của CNN:

$$J = \frac{1}{m} \sum_{i=1}^m L(\hat{y}^{(i)}, y^{(i)})$$

- Sử dụng các thuật toán tối ưu (**gradient descent* hoặc *momentum*) để tìm ra tập tham số tối ưu, giảm giá trị của J về nhỏ nhất.

7. Câu hỏi tham khảo

- Bộ lọc sau sẽ giúp phát hiện đặc trưng nào của ảnh đen trắng (gray-scale)?

$$\begin{bmatrix} 0 & 1 & -1 & 0 \\ 1 & 3 & -3 & -1 \\ 1 & 3 & -3 & -1 \\ 0 & 1 & -1 & 0 \end{bmatrix}$$

- **Phát hiện cạnh đứng**
 - Phát hiện cạnh ngang
 - Phát hiện độ tương phản của ảnh
 - Phát hiện cạnh nghiêng 45°
- Giả sử ảnh đầu vào là ảnh màu (hệ RGB) có kích thước 300×300 và không sử dụng mạng tích chập. Nếu lớp ẩn đầu tiên có 100 thành phần (units/ neurons) mỗi thành phần được liên kết đầy đủ với đầu vào, lớp ẩn này có bao nhiêu tham số (bao gồm cả bias)?
 - 9,000,001
 - 9,000,100
 - 27,000,001
 - **27,000,100**
 - Giả sử ảnh đầu vào là ảnh màu (hệ RGB) có kích thước 300×300 và sử dụng lớp tích chập có 100 bộ lọc 5×5 . Lớp tích chập này có bao nhiêu tham số (bao gồm cả *bias*)?
 - 2501
 - 2600
 - 7500
 - **7600**
 - Bạn có đầu vào kích thước $63 \times 63 \times 16$ và nhân chập nó với 32 bộ lọc kích thước 7×7 , sử dụng bước trượt $s = 2$, không padding ($p = 0$). Kích thước đầu ra là?

- **29x29x32**
- 16x16x32
- 29x29x16
- 16x16x16

5. Thực hiện padding đầu vào kích thước $15 \times 15 \times 8$ với $p = 2$ thì thu được ma trận có kích thước nào?

- **19x19x8**
- 19x19x12
- 17x17x10
- 17x17x8

6. Bạn có đầu vào kích thước $63 \times 63 \times 16$ và nhân chập nó với 32 bộ lọc kích thước 7×7 , sử dụng bước trượt $s = 1$. Để kích thước đầu ra bằng với đầu vào (same convolution), giá trị của padding p bằng bao nhiêu?

- 1
- 2
- **3**
- 7

7. Bạn có đầu vào kích thước $32 \times 32 \times 16$ và áp dụng pooling sử dụng giá trị cực đại (max pooling) với bước trượt $s = 2$ với bộ lọc $f = 2$. Kích thước của đầu ra là?

- 15x15x16
- 32x32x8
- **16x16x16**
- 16x16x8

8. Bởi vì các lớp pooling không có tham số cần huấn luyện, nó không ảnh hưởng tới việc tính đạo hàm trong bước backpropagation?

- True
- **False**

9. “**Chia sẻ tham số**” là một ưu điểm của CNN. Phát biểu sau đây là đúng về tính chất này (chọn tất cả các đáp án đúng)?

- ☐ Nó giúp các thông số đã được huấn luyện cho một tác vụ này có thể được sử dụng cho một tác vụ khác (*transfer learning*).
- ☒ Nó giảm số lượng tham số cần huấn luyện, từ đó giảm thiểu overfitting.
- ☒ Nó giúp các bộ phát hiện đặc trưng được sử dụng lại trên nhiều vị trí khác nhau của khối đầu vào.
- ☐ Nó cho phép thiết lập các tham số bằng không khi sử dụng gradient descent, từ đó tạo thành các liên kết thưa (*sparse connections*).

10. “**Liên kết thưa**” trong CNN được hiểu như thế nào?

- Một cách chuẩn hoá (Regularization) dẫn tới việc thiết lập rất nhiều tham số bằng 0 khi sử dụng *gradient descent*.
- Mỗi lớp trong CNN kết nối tới chỉ một hoặc hai lớp khác.
- **Mỗi thành phần kích hoạt (activation) trong lớp kế tiếp phục thuộc vào một số lượng nhỏ thành phần kích hoạt của lớp trước đó.**

- Mỗi lớp kết nối với tất cả các kênh của lớp trước nó.

8. Tài liệu tham khảo

- Bài viết được viết chủ yếu dựa trên khoá học Coursera-Deeplearning.ai-CNN-Week 1 (<https://www.coursera.org/learn/convolutional-neural-networks>).

Tags: [demo \(/tags#demo\)](#) [beginner \(/tags#beginner\)](#)



← **PREVIOUS POST (/2018-07-15-TRANSFER-LEARNING-BASIC/)**

NEXT POST → (/2018-07-25-DL-EDGE-1/)

1 Comment Deep Learning và ứng dụng

Login ▾

Recommend Share

Sort by Best ▾



Join the discussion...

LOG IN WITH

OR SIGN UP WITH DISQUS

Name



Pham Xuan Trung • a month ago

Tác giả cần sửa lại mục 5) CNN đơn giản. Output lớp Conv2 là 37x37x10 mà bạn lại ghi 27x27x10 làm tôi rất mất thời gian ko hiểu tại sao nó lại là 27 khi áp dụng công thức. Đến khi vào thẳng trang gốc của coursera mới thấy được câu trả lời (37 đúng theo công thức, không phải là 27 như bài đã ghi).

Đọc bài bạn dịch mà còn mấy chỗ không thống nhất, rất đau đầu. Còn 1 số mục tôi ko nêu ra nhưng vẫn còn rất nhầm lẫn.

^ | v • Reply • Share ▸

ALSO ON DEEP LEARNING VÀ ỨNG DỤNG

Deep learning cơ bản (phần 2)

2 comments • 3 months ago



LongNgo — Cảm ơn ad

Hướng dẫn sử dụng Deep Learning

2 comments • 3 months ago



Nam Tran — cảm ơn

- (/feed.xml) ● (mailto:deeplearningapplications@gmail.com)
- (https://www.facebook.com/pg/204704190126483)
- (https://github.com/orgs/dlapplications/teams/dl_ap)
- (https://www.facebook.com/groups/1601966719912937)

dl_ap • 2018

Theme by beautiful-jekyll (<http://deanattali.com/beautiful-jekyll/>)